NIST
National Institute of Standards and Technology
Technology Administration, U.S. Department of Commerce

NIST CENTENNIAL 1901-2001

DARPA

A Project in the ITL Pervasive Computing Portfolio

Applying ADLs to Assess Emerging Industry Specifications for Dynamic Discovery of Ad Hoc Network Services

Christopher Dabrowski and Kevin Mills

DARPA PI Meeting

January 31, 2001

Report Documentation Page

Form Approved
OMB No. 0704-0188

1. REPORT DATE **31 JAN 2001**	2. REPORT TYPE	3. DATES COVERED **00-00-2001 to 00-00-2001**

4. TITLE AND SUBTITLE **Applying ADLs to Assess Emerging Industry Specifications for Dynamic Discovery of Ad Hoc Network Services**	5a. CONTRACT NUMBER 5b. GRANT NUMBER 5c. PROGRAM ELEMENT NUMBER
6. AUTHOR(S)	5d. PROJECT NUMBER 5e. TASK NUMBER 5f. WORK UNIT NUMBER
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) **National Institute of Standards and Technology,Gaithersburg,MD,20899**	8. PERFORMING ORGANIZATION REPORT NUMBER
9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)	10. SPONSOR/MONITOR'S ACRONYM(S) 11. SPONSOR/MONITOR'S REPORT NUMBER(S)

12. DISTRIBUTION/AVAILABILITY STATEMENT

Approved for public release; distribution unlimited

13. SUPPLEMENTARY NOTES

DARPA PI Meeting, 31 Jan 2001

14. ABSTRACT

15. SUBJECT TERMS

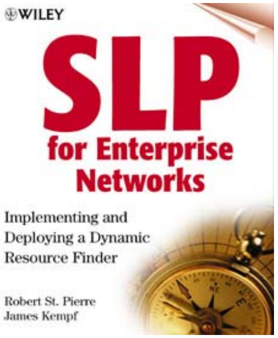
16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
a. REPORT **unclassified**	b. ABSTRACT **unclassified**	c. THIS PAGE **unclassified**	**Same as Report (SAR)**	**19**	

Project Goals

1) Use ADLs and associated tools to **analyze Discovery Protocol specifications** to assess consistency and completeness wrt dynamic change conditions—provide basis for gauges.
2) **Compare and contrast emerging** commercial service **discovery technologies** with regard to function, structure, behavior, performance and scalability in the face of dynamic change.

JINI

Universal
Plug and Play

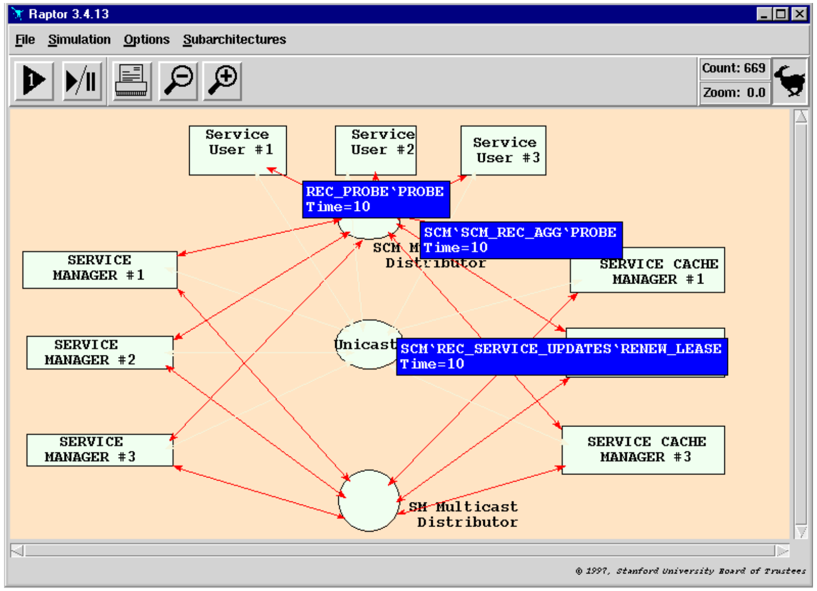


WILEY
SLP
for Enterprise Networks
Implementing and Deploying a Dynamic Resource Finder
Robert St. Pierre
James Kempf

Presentation Topics

- Planned Approach to Modeling and Analysis and Current Status
- Technical Discussion of Initial Progress
  - Generic and Specific UML Models Encompassing Jini, UPnP, & SLP
  - *Rapide* Model for Jini **(**90% complete)
- Upcoming Milestones and Planned Publications

Modeling Function, Structure, and Behavior



Objectives

(1) Provide increased understanding of the competing dynamic discovery services emerging in industry

(2) Develop metrics/gauges for comparative analysis of different approaches to dynamic discovery and for analyzing consistency and completeness of discovery protocols

(3) Assess suitability of architecture description languages to model and analyze emerging dynamic discovery protocols

Technical Approach

- Develop ADL models from selected specifications for service discovery protocols and develop a suite of scenarios and topologies with which to exercise the ADL models
- Propose a set of invariant properties that all dynamic discovery protocols should satisfy
- Propose a set of metrics, based on partially ordered sets, with which to compare and contrast discovery protocols
- Analyze the ADL models for inconsistencies, to assess invariant satisfaction, and to compare and contrast protocols

Status as of January 31, 2001

- Developed a generic UML model encompassing the structure and function of Jini, UPnP, SLP, Bluetooth, and HAVi
- Projected specific UML models for Jini, UPnP, and SLP
- Developed a Rapide Model of Jini structure, function, and behavior (90% complete)
- Drafted a scenario language to drive the Rapide Jini Model; currently being implemented.
- Developed some initial invariants and constraints for Jini behavioral model
- Discovered a number of ambiguities and inconsistencies in Jini Specification V1.1
- Discovered a major architectural issue in the interaction between Jini directed discovery and multicast discovery

Products

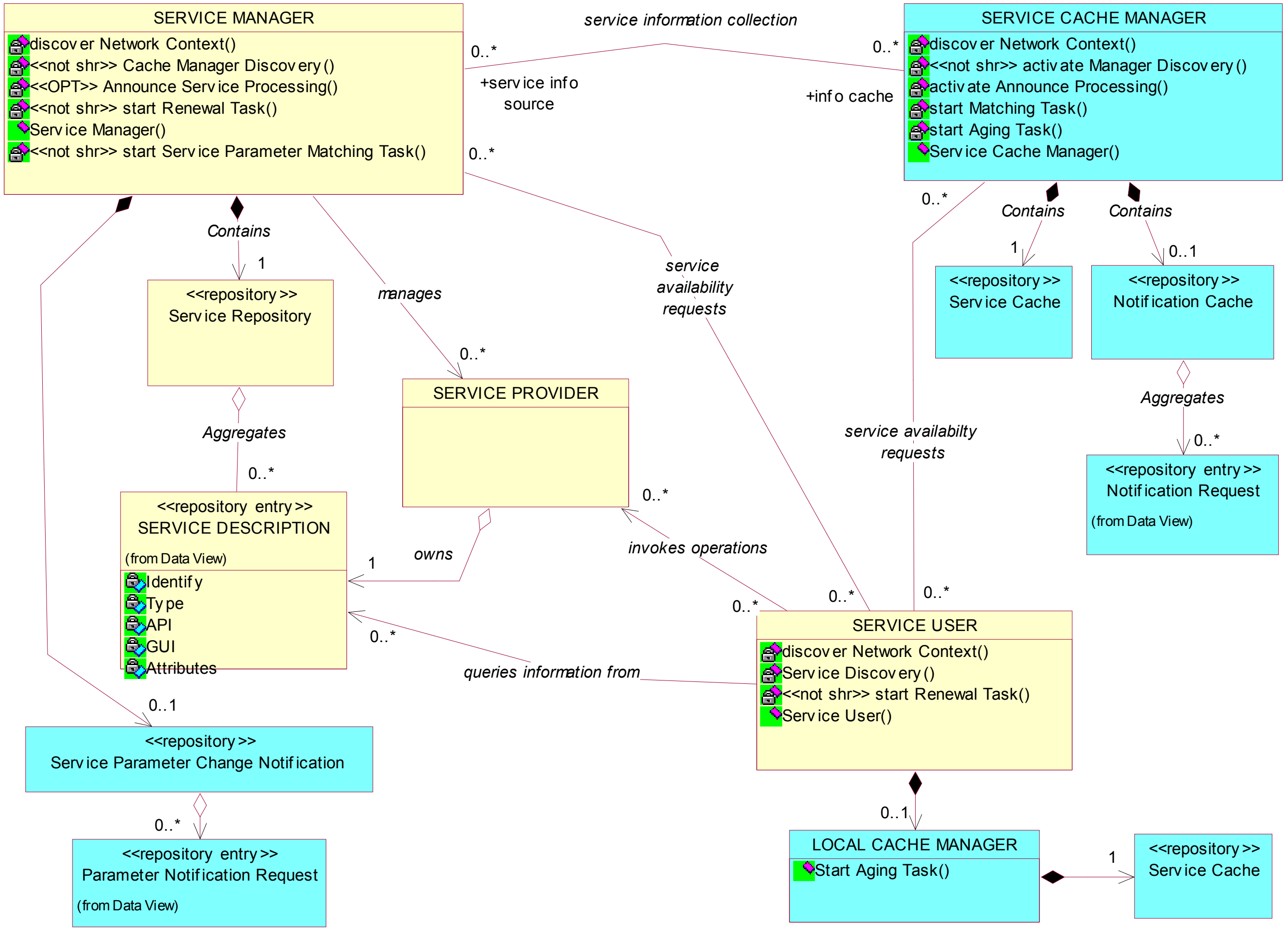
- Rapide specifications of Jini, Universal Plug and Play (UPnP), and Service Location Protocol (SLP)
- Scenarios and topologies for evaluating discovery protocols
- Suggested invariant properties for service discovery protocols
- Suggested metrics, based on partially ordered sets (POSETs), for comparing and contrasting discovery protocols
- Paper identifying inconsistencies and ambiguities in Jini and UPnP and describing how they were found
- Paper proposing invariants for service discovery protocols, and evaluating how Jini, UPnP, and SLP fare
- Paper comparing and contrasting Jini, UPnP, and SLP at the level of POSET metrics

Benefits from Using Architecture, ADLs, & Tools

- **Represent essential complexity** with effective abstractions

- Provide a framework and context
  - to more easily **pinpoint** where **inconsistencies and ambiguities** may exist within software implementing specifications & to understand how they arise
  - to **compare and contrast** different discovery **protocols** (Jini, UP&P, SLP)
  - to **define gauges** that yield qualitative and quantitative measures

Generic UML Structural Model of Service Discovery Protocols



SERVICE MANAGER
discov er Network Context()
<<not shr>> Cache Manager Discov ery ()
<<OPT>> Announce Serv ice Processing()
<<not shr>> start Renewal Task()
Serv ice Manager()
<<not shr>> start Serv ice Parameter Matching Task()

service information collection
0..*
+serv ice inf o source
+inf o cache
0..*

SERVICE CACHE MANAGER
discov er Network Context()
<<not shr>> activ ate Manager Discov ery ()
activ ate Announce Processing()
start Matching Task()
start Aging Task()
Serv ice Cache Manager()

0..*
Contains
1
<<repository>>
Serv ice Repository

manages
0..*
SERVICE PROVIDER

service availability requests

0..*
Contains
1
<<repository>>
Serv ice Cache

Contains
0..1
<<repository>>
Notif ication Cache
Aggregates
0..*
<<repository entry>>
Notif ication Request
(from Data View)

service availabilty requests

Aggregates
0..*
<<repository entry>>
SERVICE DESCRIPTION
(from Data View)
Identify
Ty pe
API
GUI
Attributes

owns
1
0..*
invokes operations
0..*
0..*
0..*
0..*
queries information from

SERVICE USER
discov er Network Context()
Serv ice Discov ery ()
<<not shr>> start Renewal Task()
Serv ice User()

0..1
<<repository>>
Serv ice Parameter Change Notif ication
0..*
<<repository entry>>
Parameter Notif ication Request
(from Data View)

0..1
LOCAL CACHE MANAGER
Start Aging Task()
1
<<repository>>
Serv ice Cache

Architecture Description Languages and Tools

Allow us to **model the essential complexity** of discovery protocols, while ignoring the incidental complexity

Jini documented in a 385 page specification; however, the document is static and thus captures only the **normative complexity** because most of the **essential complexity** arises through interactions among distributed, independently acting, Jini components.

Incidental complexity represented by the code: for example consider Core Jini – an 832 page commentary on the massive amount of Java code that comprises Jini, which also depends on complex underlying code for Remote Method Invocation, Distributed Events, Object Serialization, TCP/IP, UDP, HTTP, and Multicast Protocol Implementation.

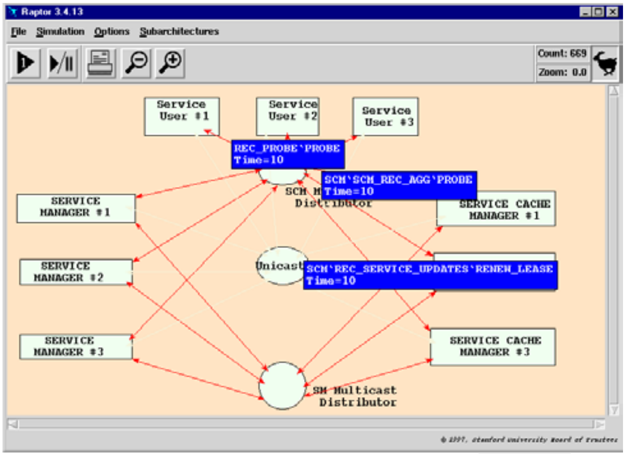
Rapide, an Architecture Description Language and Tools Developed for DARPA by Stanford

MODELING ESSENTIAL COMPLEXITY

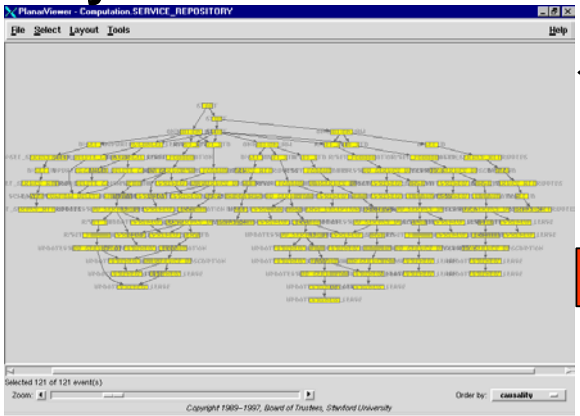
Specification of Rapide Architecture

```
-- *****************************************************
-- ** 3.3  DIRECTED DISCOVERY CLIENT INTERFACE      **
-- *****************************************************
-- This is used by all JINI entities in directed
-- discovery mode.  It is part of the SCM_Discovery
-- Module. Sends Unicast messages to  SCMs on list of
-- SCMS to be discovered until all SCMS are found.
-- Receives updates from SCM DB of discovered SCMs and
-- removes SCMs accordingly
-- NOTE: Failure and recovery behavior are not
-- yet defined and need reviw.
TYPE Directed_Discovery_Client
  (SourceID : IP_Address; InSCMsToDiscover : SCMList; StartOption : DD_Code;
   InRequestInterval : TimeUnit; InMaxNumTries : integer; InPV : ProtocolVersion)
IS INTERFACE
SERVICE DDC_SEND_DIR    : DIRECTED_2_STEP_PROTOCOL;
SERVICE DISC_MODES      : dual SCM_DISCOVERY_MODES;
SERVICE DD_SCM_Update   : DD_SCM_Update;
SERVICE SCM_Update      : SCM_Update;
SERVICE DB_Update       : dual DB_Update;
SERVICE NODE_FAILURES : NODE_FAILURES;  -- events for failure and recovery.
ACTION
 IN Send_Requests(),
    BeginDirectedDiscovery();
BEHAVIOR
   action animation_Iam (name: string);
   MySourceID        : VAR IP_Address;
   PV                : VAR ProtocolVersion;
```

Execute with Raptor Engine



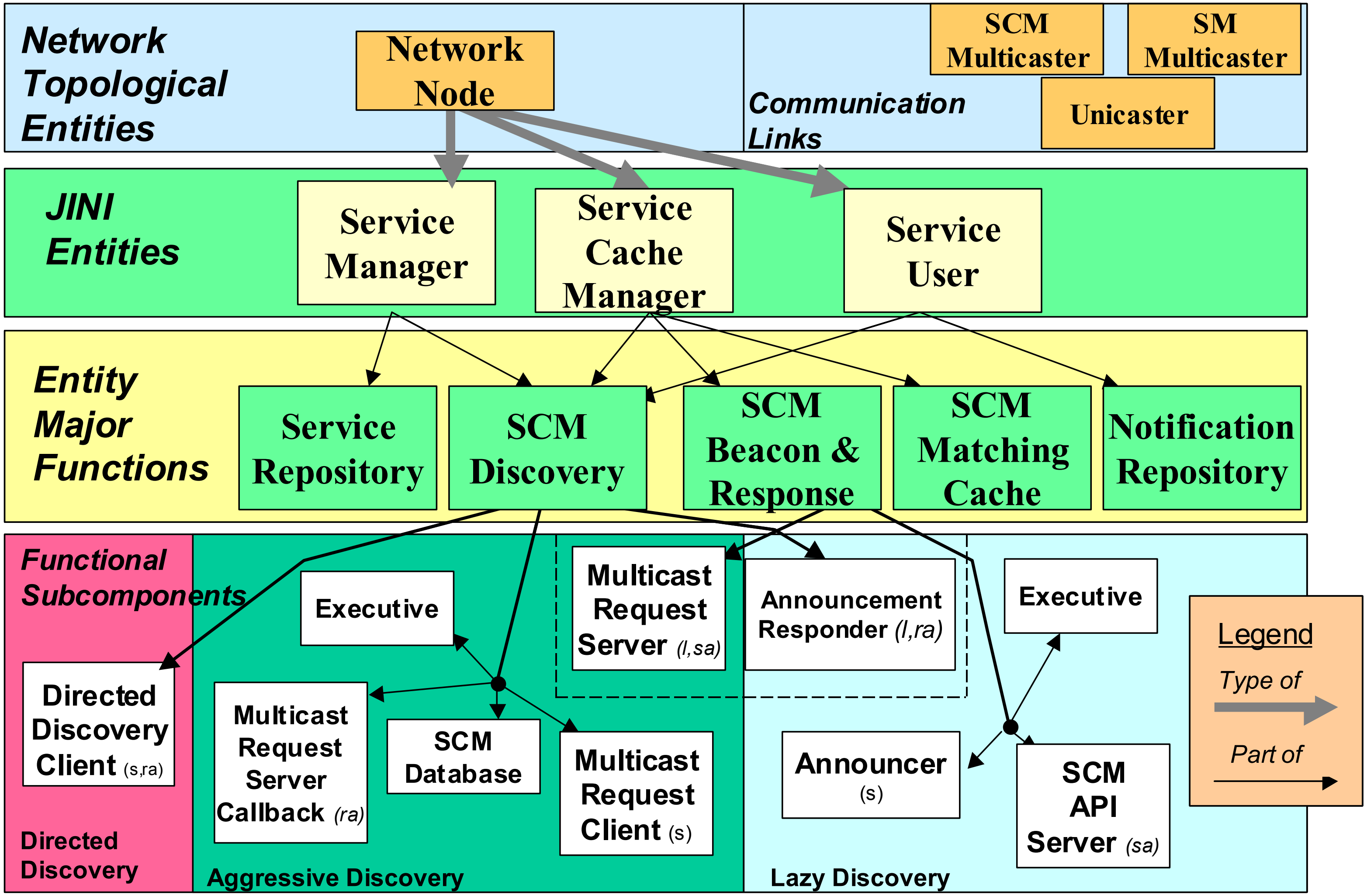
Analyze Generated POSETs


PlanaViewer - Computation SERVICE_REPOSITORY
File Select Layout Tools
Help
Selected 121 of 121 event(s)
Zoom:
Order by: causality
Copyright 1989-1997, Board of Trustees, Stanford University

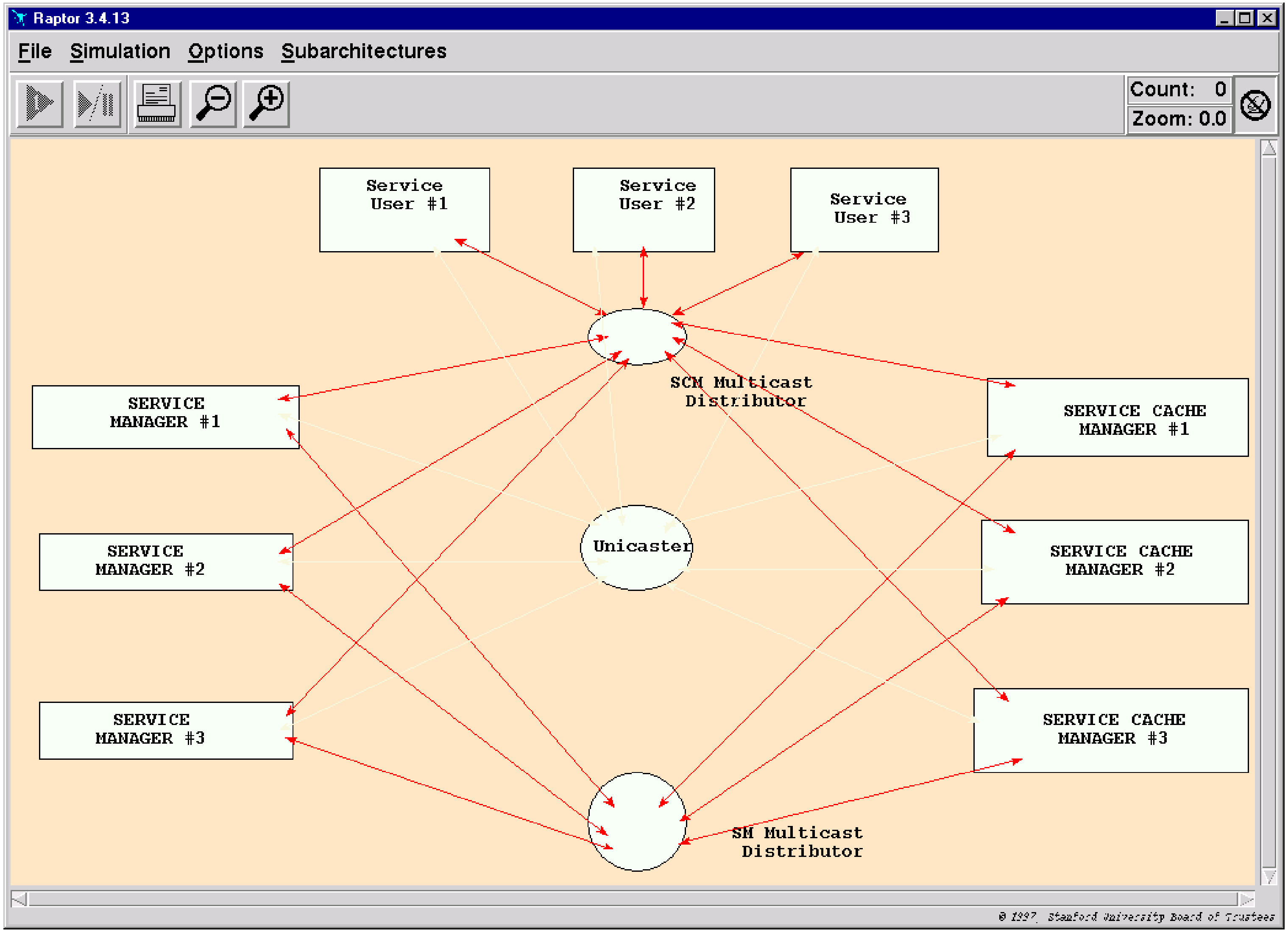
Assess Invariant Satisfaction & Constraint Violations

Layered View of Prototype JINI Architecture in Rapide

Derived from SEI Architectural Layers Approach



Execute Architecture with the Raptor Engine



Drive Model Topology with Scenarios

> StartTime {NodeFail || NodeRecover} NodeID DelayTime.

> StartTime {LinkFail || LinkRestore} NodeID DelayTime FromNode ToNode.

> StartTime {MProbeFail || MProbeRestore} NodeID DelayTime FromNode ToNode.

> StartTime {GroupJoin || GroupLeave} NodeID DelayTime.

> StartTime {AddSCM || DeleteSCM} NodeID DelayTime.

> StartTime {AddService ChangeService}NodeID DelayTime ServiceTemplate ServiceAPI ServiceGUI LeaseTime DurationTime.

> StartTime DeleteService NodeID DelayTime ServiceID.

> StartTime FindService NodeID DelayTime SMNodeID .

> StartTime AddNotificationRequest NodeID DelayTime NotificationID ServiceTemplate Transitions LeaseTime DurationTime SCMID.

> StartTime DeleteNotificationRequest NodeID DelayTime NotificationID SCMID.

Analyze Invariant→ Satisfaction & Constraint Violations in Real-Time

Sample Invariants

🕛(SM 📫 SD 📫 SCM): (SM,SD) ♍ SCM registered-services
♏ SCM ♍ SM discovered-SCMs

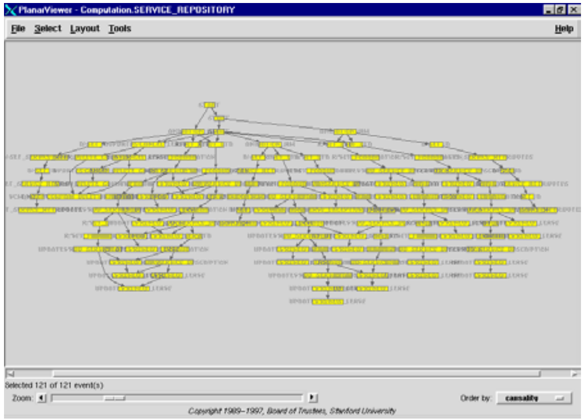
🕛(SU 📫 NR 📫 SCM): (SU,NR) ♍ SCM registered-notifications
♏ SCM ♍ SU discovered-SCMs

- SM is Service Manager
- SD is Service Description
- SCM is Service Cache Manager
- SU is Service User

- NR is Notification Request
- Registered-services is a set of (SM,SD) pairs
- Registered-notifications is a set of (SU,NR) pairs
- Discovered-SCMs is a set of SCM

→ **Invariants provide basis for defining *gauges* that provide qualitative measures of properties of a system**

Analyze POSETs→ Off-Line to Compare and Contrast Behaviors Given a Congruent Topology and Scenario



Selected 121 of 121 event(s)

Copyright 1989–1997, Board of Trustees, Stanford University

Metrics Based on Time
- Service latency?
- Service throughput?
- Recovery latency?

Metrics Based on Numbers of Messages
- Message volume?
- Message intensity?

Metrics Based on Complexity
- Degree of dependency among messages?
- Rate of constraint and invariant violations?
- Rate of exceptions?

Metrics Based on Change
- Derivative of the message intensity?
- Derivative of the service throughput?
- Derivative of the service latency?

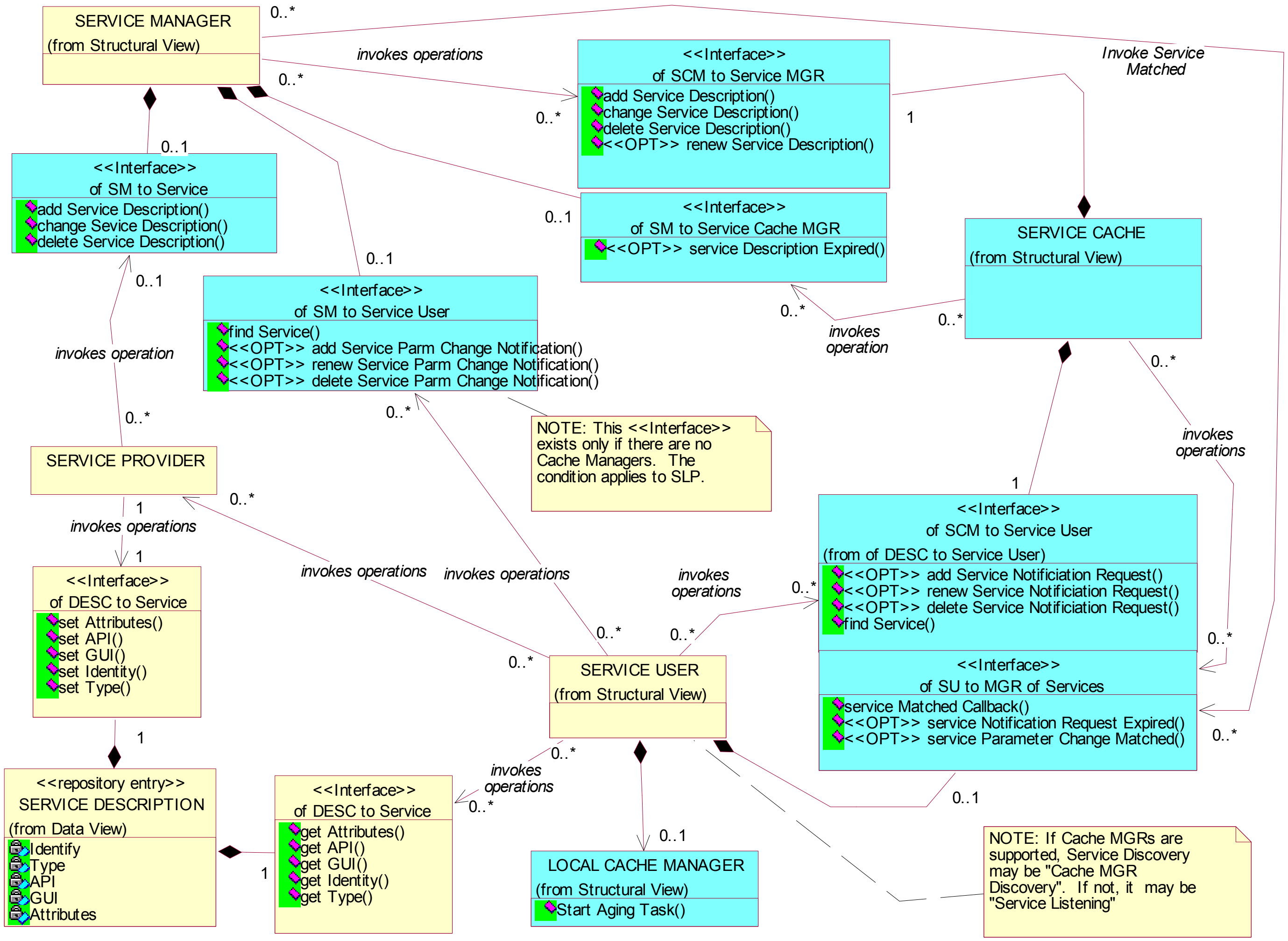
→POSET analyses provide basis for defining *gauges* that provide quantitative measures of properties of a system

Schedule and Milestones

- FY 2001
  - Operational prototype of Jini & UPnP architectures
  - Report showing initial results of analysis of Jini and compare/ contrast of Jini & UPnP; recommendations on ADLs.
- FY 2002
  - Formalization of quantitative & qualitative metrics to serve as basis for gauges; formalization of compare/contrast analysis
  - Expansion of operational prototype to incorporate metrics & resulting analysis as well as SLP (other protocols?)
  - Second report on results.

EXTRA SLIDES

Generic UML Functional Model of Service Discovery Protocols



SERVICE MANAGER
(from Structural View)

<<Interface>>
of SCM to Service MGR
- add Service Description()
- change Service Description()
- delete Service Description()
- <<OPT>> renew Service Description()

<<Interface>>
of SM to Service
- add Service Description()
- change Sevice Description()
- delete Service Description()

<<Interface>>
of SM to Service Cache MGR
- <<OPT>> service Description Expired()

SERVICE CACHE
(from Structural View)

<<Interface>>
of SM to Service User
- find Service()
- <<OPT>> add Service Parm Change Notification()
- <<OPT>> renew Service Parm Change Notification()
- <<OPT>> delete Service Parm Change Notification()

NOTE: This <<Interface>> exists only if there are no Cache Managers. The condition applies to SLP.

SERVICE PROVIDER

<<Interface>>
of SCM to Service User
(from of DESC to Service User)
- <<OPT>> add Service Notificiation Request()
- <<OPT>> renew Service Notificiation Request()
- <<OPT>> delete Service Notificiation Request()
- find Service()

<<Interface>>
of DESC to Service
- set Attributes()
- set API()
- set GUI()
- set Identity()
- set Type()

SERVICE USER
(from Structural View)

<<Interface>>
of SU to MGR of Services
- service Matched Callback()
- <<OPT>> service Notification Request Expired()
- <<OPT>> service Parameter Change Matched()

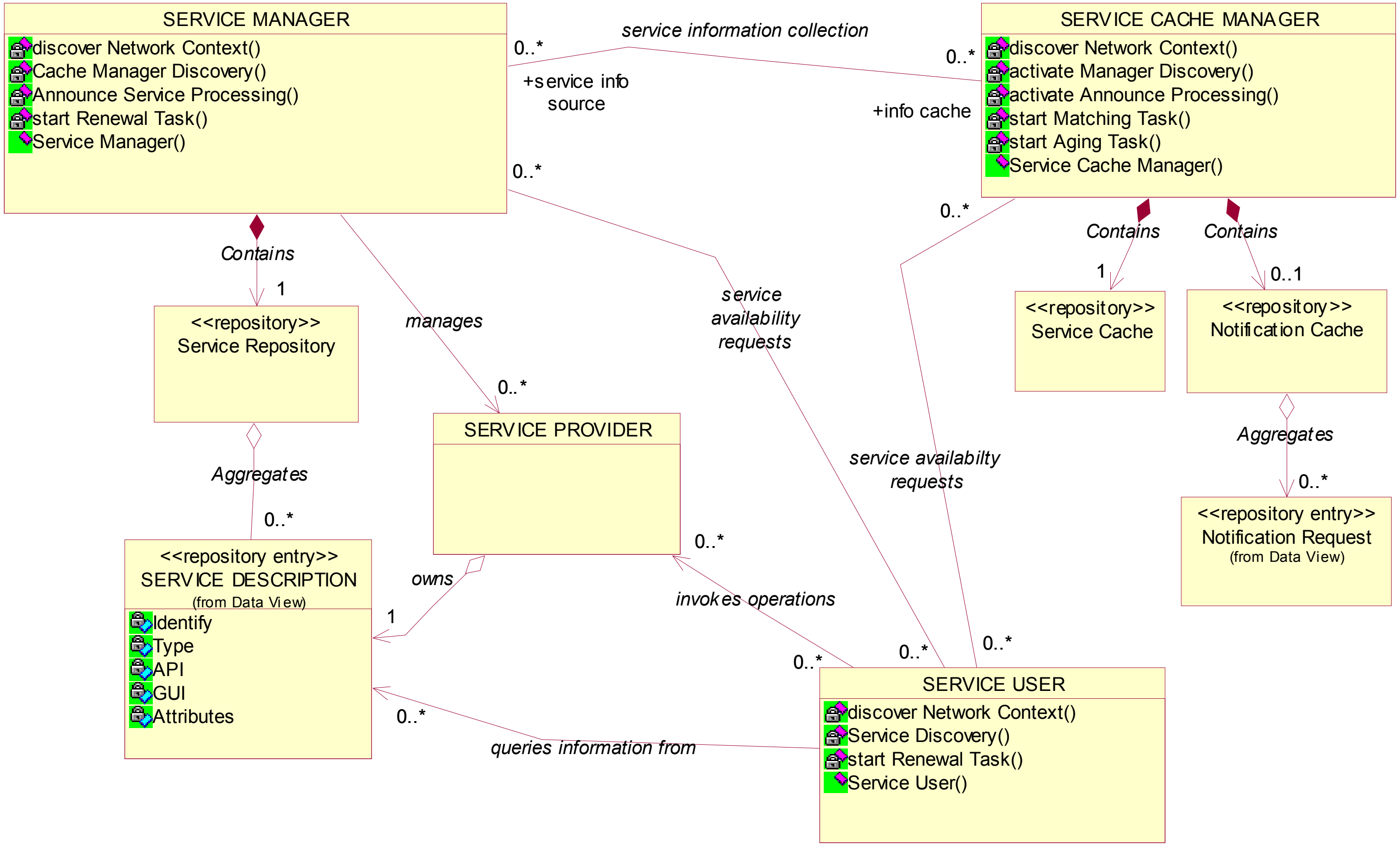
<<repository entry>>
SERVICE DESCRIPTION
(from Data View)
- Identify
- Type
- API
- GUI
- Attributes

<<Interface>>
of DESC to Service
- get Attributes()
- get API()
- get GUI()
- get Identity()
- get Type()

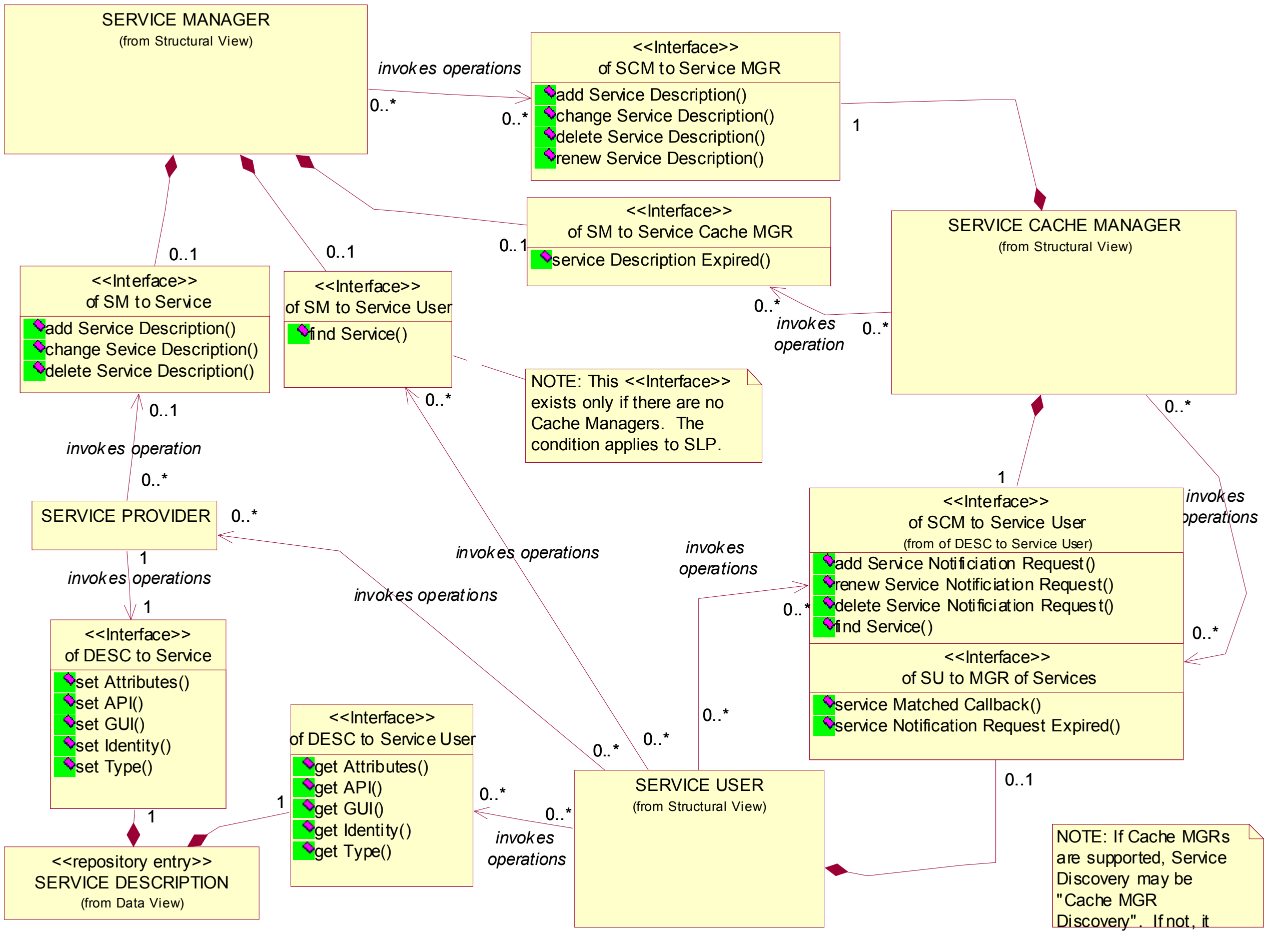
LOCAL CACHE MANAGER
(from Structural View)
- Start Aging Task()

NOTE: If Cache MGRs are supported, Service Discovery may be "Cache MGR Discovery". If not, it may be "Service Listening"

UML Structural Model of Jini



UML Functional Model of Jini



SERVICE MANAGER
(from Structural View)

<<Interface>>
of SCM to Service MGR
- add Service Description()
- change Service Description()
- delete Service Description()
- renew Service Description()

<<Interface>>
of SM to Service Cache MGR
- service Description Expired()

SERVICE CACHE MANAGER
(from Structural View)

<<Interface>>
of SM to Service
- add Service Description()
- change Sevice Description()
- delete Service Description()

<<Interface>>
of SM to Service User
- find Service()

NOTE: This <<Interface>> exists only if there are no Cache Managers. The condition applies to SLP.

SERVICE PROVIDER

<<Interface>>
of SCM to Service User
(from of DESC to Service User)
- add Service Notificiation Request()
- renew Service Notificiation Request()
- delete Service Notificiation Request()
- find Service()

<<Interface>>
of SU to MGR of Services
- service Matched Callback()
- service Notification Request Expired()

<<Interface>>
of DESC to Service
- set Attributes()
- set API()
- set GUI()
- set Identity()
- set Type()

<<Interface>>
of DESC to Service User
- get Attributes()
- get API()
- get GUI()
- get Identity()
- get Type()

SERVICE USER
(from Structural View)

<<repository entry>>
SERVICE DESCRIPTION
(from Data View)

NOTE: If Cache MGRs are supported, Service Discovery may be "Cache MGR Discovery". If not, it

Plan to Assess Scalability

- Use Rapide Models as a Basis to Construct Simulation Models for Jini, UPnP and SLP, Possibly using JavaSim (from Ohio State University) or SSFnet (from Rutgers)

- Use Results from Measurement Portion of the Project to Parameterize the Simulation Models of the Discovery Protocols

- Design Experiments to Assess the Effect of Large Service and Device Populations on Network Traffic